Summer 2019 Project 8: Strategy Learner

By Nan Mao (nmao7)
GT ID: 903363914

- Describe the steps you took to frame the trading problem as a learning problem for your learner. What are your indicators? Describe how you discretized (standardized) or otherwise adjusted your data. If not, tell us why not.

  BagLearner & RTLearner

  In this project, I created a strategy using BagLearner implemented with RTLearner. Because the learner I choose, I didn't discretize data. Unlike Q-Learner, RTLearner doesn't need to discretize or standardize data. It accepts daily price and all numeric indicator features. To avoid overfitting in-sample data, I set leaf size = 5. I used mode rather than mean for leaf value selection in RTLearner. I set bag size 20 to ensemble random tree learners.

  Indicators

  I used four indicators as the same used in Manual Strategy assignment.

  1. Volatility indicator: It calculated as Standard Deviation of daily prices.
  2. SMA & price/SMA indicator: SMA is calculated by rolling mean of daily prices during look back window frame, which is 10 days in this assignment.
  3. Bollinger Band indicator: It plotted two lines at 2 standard deviations of daily return on either side of simple moving average line.
     Upper band = SMA + 2 * standard deviation of daily prices
     Lower band = SMA - 2 * standard deviation of daily prices
     BB = (daily price – Price Average)/2 * STD
  4. EMA & price/EMA indicator: Compared to SMA, EMA gives more weights to the recent prices. So EMA is more sensitive to the recent price change.
     EMA = Price(today) * multiplier + EMA(yesterday) * (1− multiplier), where multiplier is the weight of EMA and calculated as 2/(look back window + 1)

  Steps

  In training method addEvidence(), with given symbol and training period, I applied util.get_data() function to get daily prices for the symbol and got four indicator values through functions in indicators.py. These are the features X for learner. I calculated daily return for the symbol during the training period. For each day, if the daily return is greater than impact rate plus 0.005, I decided to make a long trade. If the daily return is less than negative impact rate minus 0.005, I decided to make a short trade. If the daily return falls in between, I did nothing. Clearly, this is a classification problem with Y label has 3 different values. +1 means long/buy position. -1 mean short/sell position and 0 means holding. Below is Pseudo code to explain how do I label Y values.

```
if daily price > impact + 0.005:
        Long position: label Y = 1
elif daily price < - impact – 0.005:
        Short position: label Y = -1
else:
        Holding position: label Y = 0
```

Finally, I trained prepared feature X and label Y in learner.

In testing method testPolicy(), with given symbol and testing period, I got daily prices and for indicator values same as in training method. Using test features X into learner query() function to get predicated label Y values. I created my trading dataframe with these predicated Y values. For each day, if label shows long position, I will buy shares. If label shows short position, I will sell shares. Otherwise, I will do nothing. I set holding to count the number of shares I hold and initialized it 0. With the constrains of number of shares I can hold, I used the following table to calculate the number of shares I can buy/sell for each trade.

holding	-1000	0	1000
BUY	2000	1000	0
SELL	0	1000	2000

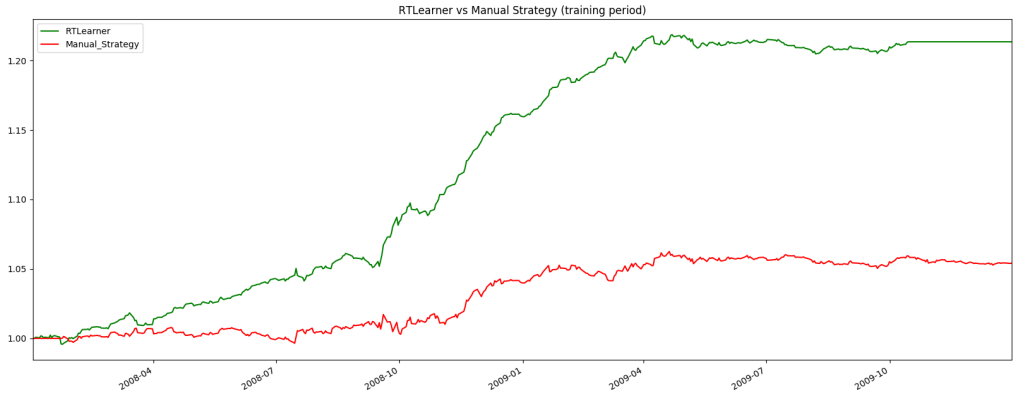
Finally testing method returns a trade dataframe with one column as number of shares traded. If the number is positive, it means long/buy. If the number is negative, it means short/sell the stocks.

- Experiment 1:

  I used same four indicators for both Learner Strategy and Manual Strategy and traded JPM during the training period. For better comparison, both strategies use 0 commission fee and 0.005 impact rate.

```
======Experiment 1 report====== ======Manual Portfolio sample period=======
Cumulative Return: 0.20893161                Cumulative Return: 0.05404632
Standard Deviation of Return: 0.00149853     Standard Deviation of Return: 0.001558
Average Daily Return: 0.00042191             Average Daily Return: 0.00010886
Experiment1 Portfolio Value: 1209164.15      Manual Portfolio Value: 1054262.45
```

The table compares the portfolio statistics for each strategy. Learner strategy has much higher cumulative return and average daily return. Standard deviations are close for both strategies. Learner strategy has a higher portfolio value at the end of the period.

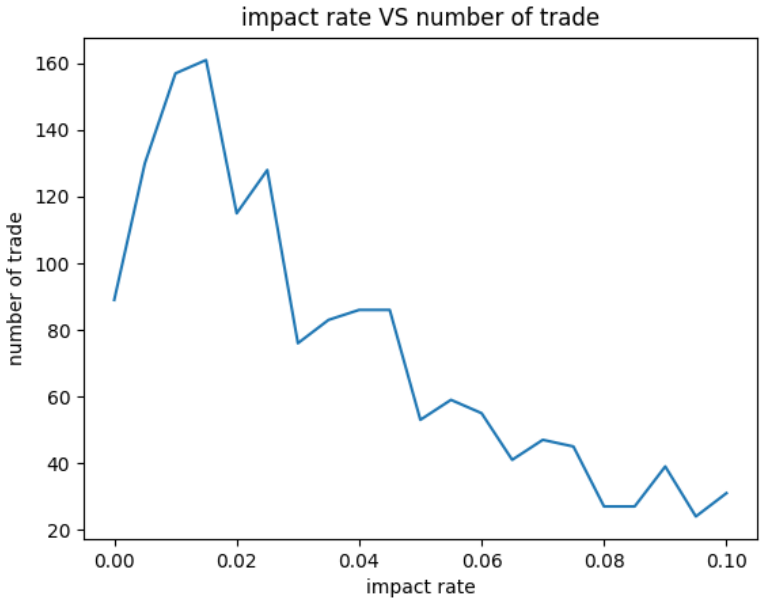


The plot above shows the outcome of the comparison. I used compute_portvals() function in marketsimcode.py to calculate the portfolio value under each trading strategy. The plot shows the portfolio values under each strategy from January 1, 2008 to December 31, 2009. Red line represents learner strategy and green line represents manual ruled strategy. It clearly shows that learner strategy learned better and better and has much better performance than manual rule strategy.

For in sample data, I expect similar result from learner strategy and same result from manual rule strategy. For manual rule strategy, same data with same criteria will generate same result each time. For learner strategy, random tree learner selects the feature randomly and ensemble bag learner selects data instance randomly, so the result may be little bit different for each time. Generally, random tree with ensemble learner will generate pretty close result after learning.

- Experiment 2:

  My hypothesis regarding how changing the value of impact should affect in sample trading behavior and results is as impact rate increased, the chance of trading will decrease and the portfolio value may increase first and then decrease. It is because I used impact rate to compare with daily return and labeled the Y value. As impact rate increases, there will be less daily return higher than the impact rate, so there will generate less trades. When impact rate is low, increasing impact rate may help to constrain the possible negative profit trades. However, when impact rate is high, only few trades happened, which reduces the chance of profit making trades. When impact rate is higher than all daily return, there will be no trade happen and strategy can't make any profit at all. And this is the reason I compared daily return with impact rate plus a small number when I decided my trades.



Metrics #1. Number of trades

The plot above shows the number of trades happened as impact rate increases. The general trend is as impact rate increases, the number of trades increases at the beginning and then decreases gradually. This graph partially supports my hypothesis. However, as impact rate increases from 0.0 to 0.01, the number of trades increases.

Metrics #2. Portfolio value

The plot below shows the relationship between impact rate and portfolio ending values. The general trend is portfolio ending values decreases as impact rate increases. However, the max JPM portfolio value appears when impact rate is about 0.01. This graph supports my hypothesis as well.